



NACKADEMIN

Inlämning IoT22- Inbyggda system: arkitektur och design

2023-05-01

Björn Larsson

Introduktion

Rapporten beskriver min upplevelse och hur jag gått tillväga i stora drag vid utvecklingen av en drivrutin för UART-kommunikation på STM32F411x-plattformen. Rapporten bör ge läsaren en insikt i processen att utveckla enklare drivrutiner som nybörjare inom området.

Förutom det kommer också en drivrutin för kontroll av LEDs att utvecklas. Detta som ett exempel på hur man kan använda UART-kommunikation på mikrokontrollern för att styra andra enheter.

Längst ner i denna rapport länkar jag till projektet på min Github där samtliga filer finns uppladdade. Eftersom vi inte haft tillgång till fysisk hårdvara, simulering eller labb-miljö kommer inga direkta tester att utföras.

Flytande text


Jag började med att lägga mycket fokus på att bli bekant med dokumentationen för projektet som i praktiken innebar följande 3 dokument:

1xDatasheet

1xUser manual

1xReference manual

Till en början låg mitt fokus på att bekanta mig med ovan dokumentation för projektet men jag insåg snabbt att texterna var för matiga att börja med och gick istället in på diverse youtube klipp som lärare och elever länkat till för att få en bättre helhetsbild. När man är ny inom områden som dessa inser man snabbt att allt tar mycket längre tid än man tror, och att man behöver lägga tid på saker som indirekt har med uppgiften i sig att göra. Detta var fallet för mig med STM32CubeIDE och jag la således mycket tid på att



bli någorlunda bekväm med detta verktyg. Efter att ha läst igenom datablad, lyssnat på lärare osv blev det också tydligt att jag behövde ha en djupare förståelse för hur de hexadecimala & binära talsystemen fungerar så jag valde även att lägga mer tid där. Det blev snabbt tydligt varför det är så fördelaktigt att använda sig av hexadecimala systemet istället decimaltal eller att skriva långa rader med ettor och nollar osv. Efter att detta blivit tydligare och i takt med att vi börjat kolla mer på själva koden insåg jag även att min kunskap om bitvisa operatorer i C blivit lite rostiga då vi inte lagt någon större vikt på dessa i föregående kurser, så därav blev det nästa steg för mig. Jag gick igenom de vanligaste operatorerna som bitwise and, or, not, xor och gjorde även ett par övningar med left\right shift.

När jag väl börjat känna mig tillräckligt bekväm ovan nämnda saker gick jag tillbaka till att titta på dokumentation och IDE:n samt började fokusera på att få en bättre förståelse för UART-protokollet. Jag gick igenom varje register med beskrivning av dess funktioner för att få en bättre förståelse av hur de används i koden. Här började jag också med att kommentera UART koden med egna ord och med hjälp av dokumentation samt notiser från lektionstid. Det är nästan här jag lagt mest tid och upplevt det som mest givande då man verkligen fått många aha upplevelser och lärt sig mycket nytt.

Jag har även haft sessioner med ett par studenter där vi mest suttit och pratat runt koden och hur vi tolkat de olika funktionerna samt vart i dokumentationen man kan läsa mer om diverse områden.

Eftersom det var ett tag sedan vi läste C och att man snabbt glömmer när det inte används så har jag även lagt mycket tid på att återaktivera gamla minnen som t.ex. hur man skriver egna header filer och inkluderar dessa, macro-funktionalitet samt hur man bryter isär kod till egna source filer.

Jag har också beställt hem ett stm32f407 discovery kort som jag ser fram emot att labba med för att vidareutvecklas på egen hand nu när kursen är över.

Jag upplevde uppgiften till en början som oklar och förvirrande men efter att det tydliggjordes tycker jag i slutändan att den varit väldigt lärorik och att man fått mycket nytt att bita i.

Resultat

Som jag nämnt ovan har jag inte kunnat testa koden fullt ut då förutsättningarna för detta saknats, men vad jag har gjort är att säkerställa att koden gått att kompilera utan varningar och felmeddelanden i CubeIDE.

Länk till min github med projektfiler: <https://github.com/wamurti/Projekt1>